ISTANBUL KULTUR UNIVERSITY

Department of Computer Engineering

2018-2019, 2nd Semester

CSE0440: Artificial Intelligence

8-Puzzle Solver Report

Statistics and Comparison of Algorithms

Ferhat Erduran

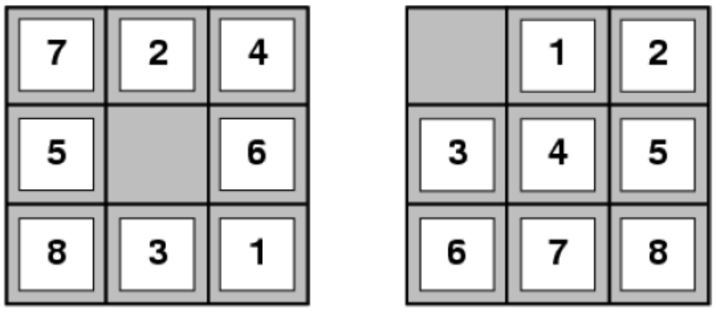
1600001092

Section B

Doç. Dr. Zeynep Orman

Instructor

1. Overview:



Start State Goal State

- Consists of a 3x3 board with 8 numbered tiles and a blank space.
- A tile adjacent to the blank space can slide into the space.
- The object is to reach a specified goal state, such as the one shown on the right.

Implement an agent to solve an 8-puzzle with the following search methods:

- Breadth-first
- Depth-first
- Depth limited (depth limit must be an argument, try different depth limits)
- Iterative deepening search

2. Algorithms:

2.1. Breadth-First Search (BFS)

BFS was able to find a solution with *length* 26, but compared to the DFS variants, it used more memory.

For the 26-path solution, BFS expanded **4,536,889** nodes and the maximum size of the fringe was **8,086,899**. The solution took **23.687** seconds and used **129.39 MB** of memory.
(Screenshot: "BFS.png")

2.2. Depth-First Search (DFS)

DFS was not able to find a solution to the attempted problems, always ending up in an infinite loop. Therefore, its implementation was removed from the agent. (Note that the only difference between it and DLS is the user-defined *limit*. The rest of the implementation is the same.)

3.3. Depth-Limited Search (DLS)

DLS was able to find a solution with *length* and *limit* 26, which was the smallest *limit* that found a solution. However, DLS was able to find a much optimal solution with *length* and *limit* 28.

For the 26-path solution, DLS expanded **3,837,792** nodes and the maximum size of the fringe was **6,914,181**. The solution took **21.572** seconds and used **110.63 MB** of memory. (Screenshot: "DLS2.png")

For the 28-path solution, DLS expanded **60,136** nodes and the maximum size of the fringe was **108,464**. The solution took **0.308** seconds and used **1.74 MB** of memory. (Screenshot: "DLS3.png")

3.4. Iterative Deepening Search (IDS)

IDS's performance was lower than that of DLS, but this was due to the fact that the limits given to DLS were precise and correct. In a scenario where the appropriate limits are unknown, IDS will always be able to find a solution, whereas DLS may fail if the limit is smaller than the minimum solution depth. With an initial *limit* of 5 and an increasing *factor* of 5, IDS found a *length* 28 solution. With an initial *limit* of 14 and an increasing *factor* of 3, IDS found a *length* 26 solution.

For the 26-path solution, IDS expanded **4,884,308** nodes and the maximum size of the fringe was **8,677,049**. The solution took **27.523** seconds and used **138.83 MB** of memory. (Screenshot: "IDS2.png")

For the 28-path solution, DLS expanded **2,896,741** nodes and the maximum size of the fringe was **4,877,119**. The solution took **15.491** seconds and used **78.03 MB** of memory. (Screenshot: "IDS.png")

Note that these test inputs may not be optimal.

3. Conclusion

In conclusion, we can put these algorithms in the following order from least optimal to most:

DFS -> BFS -> IDS -> DLS

However, it should again be noted that IDS will be more optimal than DLS in situations where an appropriate limit may not be initially designated.